Mini-lab 1

Ying Lin

10/24/2019

Week 1

Q1. Visualize relation between stimulus, choice, and spike rates

We want to visualize 20 cells given this large dataset. To extract 20 cells, we created a subset of the data by filtering and ramdomly sampling ID, which provides information on the animal, the cell, and the run. We used the `mutate` and `filter` function to create the subset which is stored under *d.cells*.

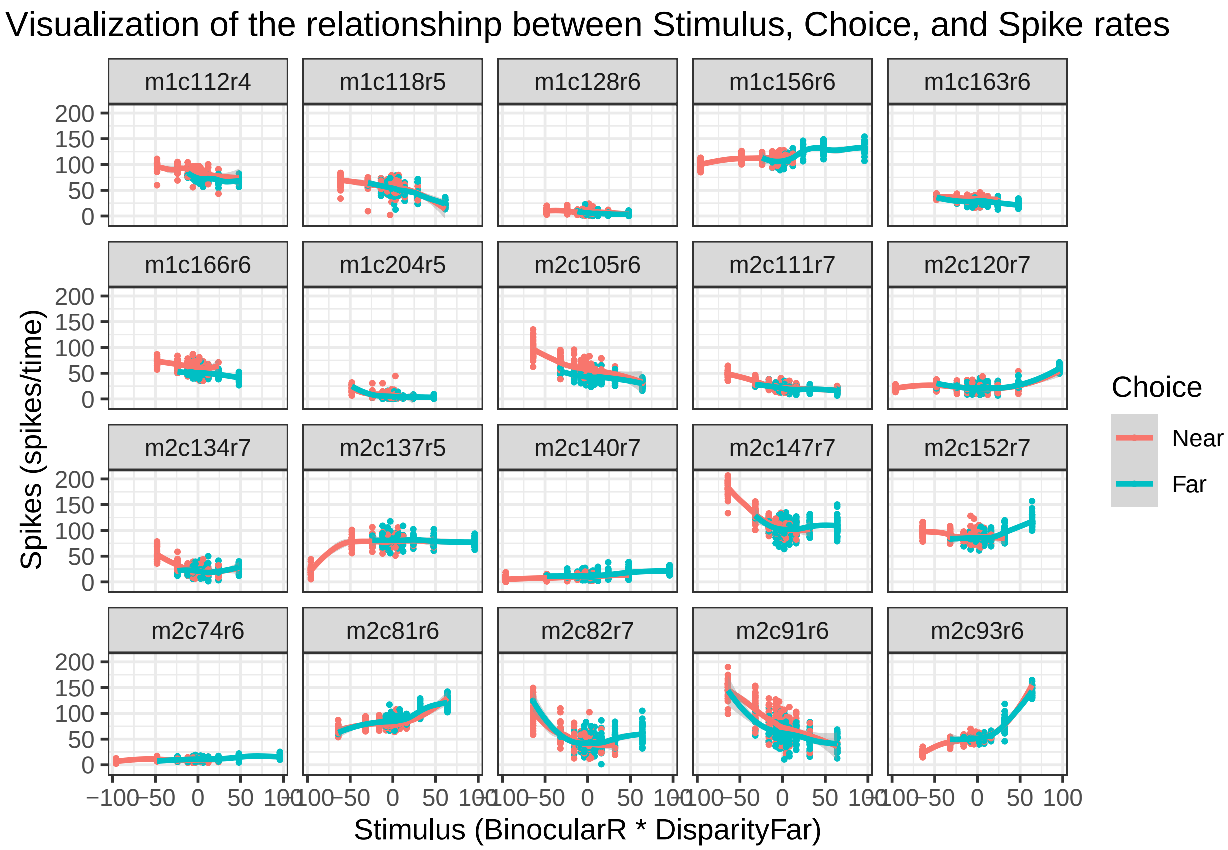
We then used ggplot2 to visualize the relationship between Stimulus (x-axis) and Spikes (y-axis) based on the different choices (near vs. far)

In the figure, there are 20 panels in total and each of the panel indicates the relationship of Stimulus on Spikes for each specific *monkey*, *cell*, and *run*.

We can see that the effect of Stimlulus on Spikes does not seem to be linear for most cells.

```r
# make subset data for 20 cells
set.seed(100)
d.cells = spikes %>%
  mutate(
    Choice = as.character(as.numeric(Choice))) %>%
  filter(BinocularR!=0,
         ID %in% sample(levels(.$ID), 20))

# plot data
ggplot(d.cells, aes(x = Stimulus, y = Spikes, color = Choice, )) +
  geom_point(size = 0.5) + facet_wrap(~ID, ncol=5) +
  geom_smooth() +
  scale_x_continuous("Stimulus (BinocularR * DisparityFar)") +
  scale_y_continuous("Spikes (spikes/time)") +
  scale_color_discrete(name = "Choice", labels = c("Near", "Far")) +
  labs(title="Visualization of the relationshinp between Stimulus, Choice, and Spike rates") +
  theme(plot.title = element_text(hjust=0.5))
```

```
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Q2. Linear Model

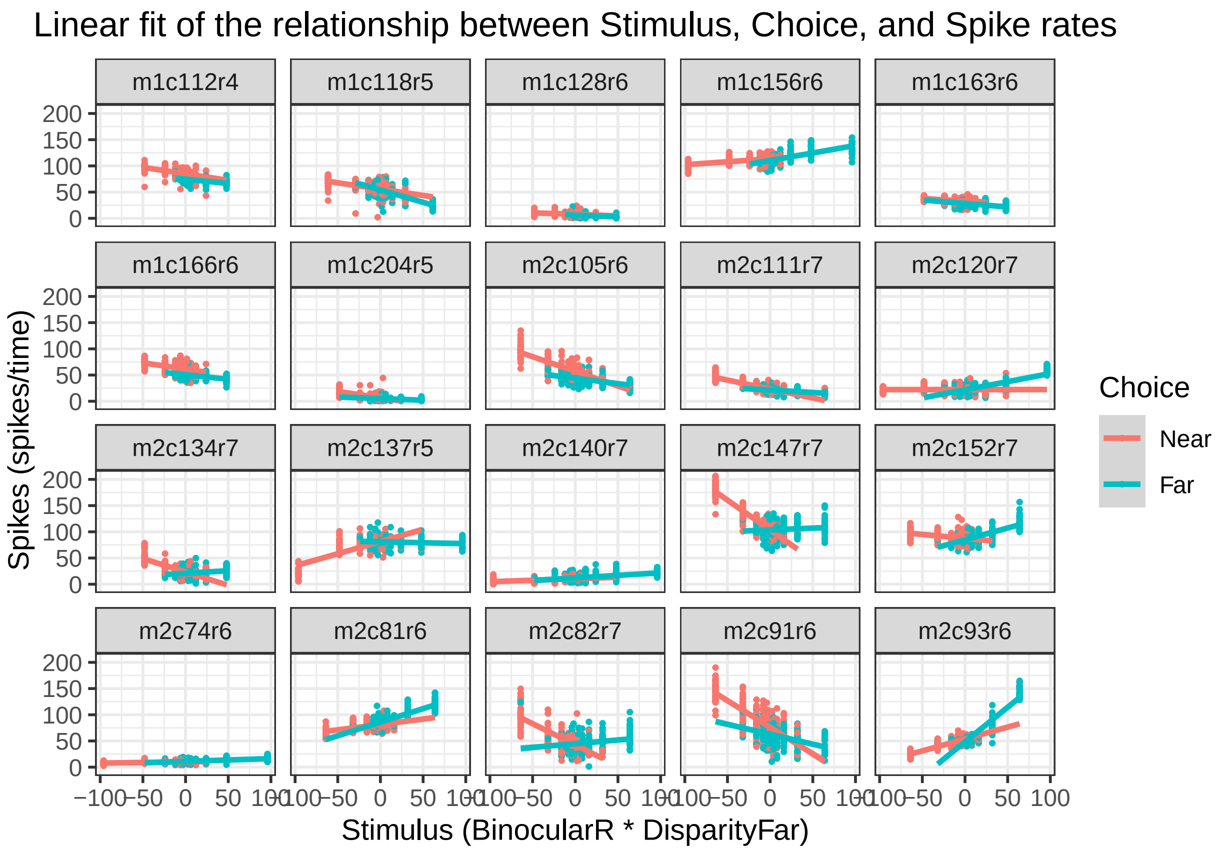
From the previous figure, we observed that the effect of Stimulus on Spikes does not seem to be linear for most cells. By adding a linear model to the data it forces the fit to be more linear, this is because the model assumes the dependence of spikes on stimulus and choice to be linear.

To define the linear model, we grouped our subdataset (d.cells) by ID and defined the linear model for spikes as $Spikes = Stimulus + Choice$.

Using `geom_smooth` in ggplot2 we defined the `method` to equal to `lm` to indicate a linear model. From the output figure we can see that trend lines are more straight compared to the previous figure where the trend lines are more curved.

```
# define lm model
lm_model =
  d.cells %>%
  group_by(ID) %>%
  do(model = lm(Spikes ~ 1 + Stimulus + Choice, data = .))

# plot lm model
ggplot(d.cells, aes(x = Stimulus, y = Spikes, color = Choice)) +
  geom_point(size = 0.5) + facet_wrap(~ID, ncol=5) +
  geom_smooth(method = 'lm') +
  scale_x_continuous("Stimulus (BinocularR * DisparityFar)") +
  scale_y_continuous("Spikes (spikes/time)")+
  scale_color_discrete(name = "Choice", labels = c("Near", "Far")) +
```

```r
  labs(title="Linear fit of the relationship between Stimulus, Choice, and Spike rates") +
  theme(plot.title = element_text(hjust=0.5))
```



Q3. Partial correlations

To examine the partial correlations from the linear model, we used the `glance` and `tidy` function to view the parameters. `glance` provides the goodness-of-fit for the data while `tidy` gives us the coefficents, t-stats, and p-values where we can observe whether there are any significance of Spikes with Stimulus and Choice as predictors for the 20 cells.

We can use t-statstics of *Choice* as an estimation of partial correlation with *Spikes* because t-statistics is extracted from using both estimate (*slope*) and standard error (*variability*); specifically, *t-statistics* = *estimate/se*

```r
# parameters
lm_model %>% droplevels() %>% glance(model)
```

```
# A tibble: 20 x 12
# Groups:   ID [20]
   ID    r.squared adj.r.squared sigma statistic  p.value    df logLik
   <fct>     <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl>
 1 m1c1~     0.519         0.515  8.33     128.  2.03e- 38     3  -848.
 2 m1c1~     0.495         0.492 10.6      181.  1.89e- 55     3 -1406.
 3 m1c1~     0.185         0.178  3.98      26.2 5.69e- 11     3  -651.
```

```
 4 m1c1~    0.383         0.379   9.60      86.0 8.59e- 30     3 -1029.
 5 m1c1~    0.450         0.448   4.80     195.  1.04e- 62     3 -1433.
 6 m1c1~    0.512         0.510   8.62     190.  5.00e- 57     3 -1299.
 7 m1c2~    0.307         0.301   5.69      52.6 1.28e- 19     3  -756.
 8 m2c1~    0.661         0.659  11.4      464.  1.15e-112     3 -1849.
 9 m2c1~    0.564         0.562   6.53     308.  1.34e- 86     3 -1580.
10 m2c1~    0.282         0.279  10.5       93.7 4.62e- 35     3 -1810.
11 m2c1~    0.217         0.212  11.6       49.3 1.20e- 19     3 -1392.
12 m2c1~    0.267         0.261  16.6       43.1 1.07e- 16     3 -1014.
13 m2c1~    0.339         0.336   5.63     112.  4.61e- 40     3 -1387.
14 m2c1~    0.376         0.373  20.6      144.  1.38e- 49     3 -2132.
15 m2c1~    0.0966        0.0928 12.4       25.5 2.99e- 11     3 -1887.
16 m2c7~    0.316         0.313   3.04     110.  4.36e- 40     3 -1213.
17 m2c8~    0.623         0.621   9.61     394.  1.11e-101     3 -1766.
18 m2c8~    0.152         0.147  21.7       36.3 3.02e- 15     3 -1842.
19 m2c9~    0.590         0.589  21.1      344.  3.87e- 93     3 -2143.
20 m2c9~    0.742         0.741  14.7      685.  5.42e-141     3 -1971.
# ... with 4 more variables: AIC <dbl>, BIC <dbl>, deviance <dbl>,
#   df.residual <int>
```

```r
lm_model %>% droplevels() %>% tidy(model, parametric = T)
```

```
# A tibble: 60 x 6
# Groups:   ID [20]
   ID       term        estimate std.error statistic   p.value
   <fct>    <chr>          <dbl>     <dbl>     <dbl>     <dbl>
 1 m1c112r4 (Intercept)  85.3       0.828     103.   6.89e-199
 2 m1c112r4 Stimulus     -0.209     0.0283     -7.41 2.27e- 12
 3 m1c112r4 Choice1     -10.1       1.28       -7.91 9.69e- 14
 4 m1c118r5 (Intercept)  54.4       0.780      69.7  1.34e-214
 5 m1c118r5 Stimulus     -0.346     0.0224    -15.4  7.23e- 42
 6 m1c118r5 Choice1      -2.37      1.29       -1.83 6.74e-  2
 7 m1c128r6 (Intercept)   7.50      0.357      21.0  2.48e- 55
 8 m1c128r6 Stimulus     -0.0601    0.0134     -4.50 1.07e-  5
 9 m1c128r6 Choice1      -1.60      0.622      -2.57 1.08e-  2
10 m1c156r6 (Intercept) 116.        0.932     124.   7.88e-245
# ... with 50 more rows
```

Week 2

Q5. Non-linear fit using Generative Additive Model (GAM)

The GAM model may be a better model to help us fit the data given that our observation of the 20 cells mostly shows a non-linearlity relationship. We grouped our subdataset (d.cells) by ID and then defined the GAM model for Spikes as *Spikes = s(Stimulus) + Choice*; where *s(Stimulus)* is our link function.

Again we can use the `glance` and `tidy` function to examine the goodness of fit and model parameters.

Visually, we can see that the GAM model fits our data much better.

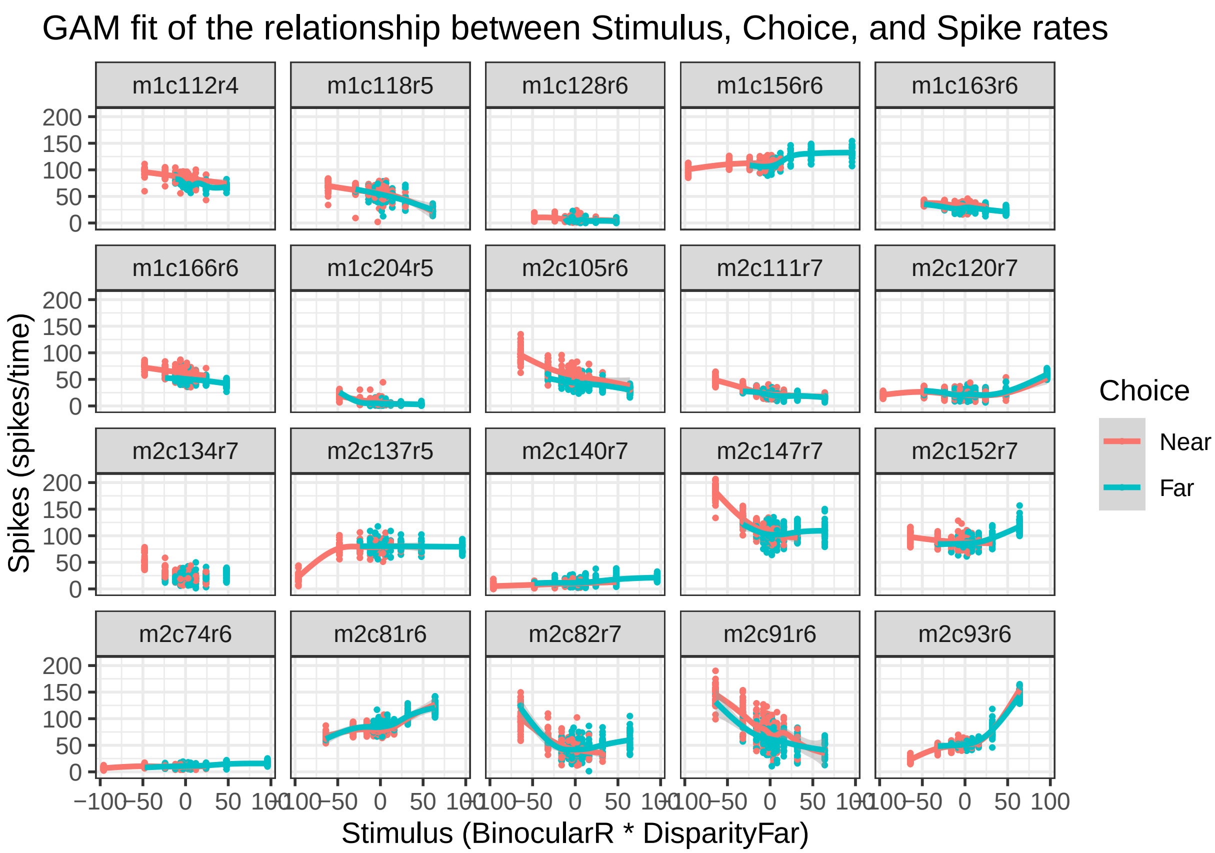
```r
# defining GAM model
gam_model =
  d.cells %>%
  group_by(ID) %>%
  do(model = gam(Spikes ~ 1 + s(Stimulus) + Choice, family = gaussian(identity), data = .))

gam_model %>% droplevels() %>% glance(model)
```

```
# A tibble: 20 x 7
# Groups:   ID [20]
   ID           df  logLik   AIC   BIC deviance df.residual
   <fct>     <dbl>   <dbl> <dbl> <dbl>    <dbl>       <dbl>
 1 m1c112r4  7.46   -842. 1700. 1729.   15605.         233.
 2 m1c118r5  4.81  -1387. 2786. 2809.   37790.         367.
 3 m1c128r6  3.     -651. 1310. 1324.    3645.         230.
 4 m1c156r6  9.15   -987. 1994. 2031.   18893.         271.
 5 m1c163r6  6.82  -1425. 2866. 2899.   10664.         473.
 6 m1c166r6  3.06  -1299. 2606. 2622.   26833.         361.
 7 m1c204r5  5.75   -736. 1485. 1509.    6467.         234.
 8 m2c105r6  5.79  -1792. 3598. 3626.   49162.         474.
 9 m2c111r7  5.93  -1460. 2934. 2963.   12320.         474.
10 m2c120r7  7.66  -1586. 3189. 3225.   20798.         472.
11 m2c134r7  5.50  -1286. 2584. 2610.   26667.         355.
12 m2c137r5  7.12   -914. 1845. 1873.   28650.         233.
13 m2c140r7  6.26  -1379. 2773. 2803.   13440.         435.
14 m2c147r7  6.08  -1919. 3852. 3881.   83318.         474.
15 m2c152r7  7.81  -1760. 3538. 3575.   43035.         472.
16 m2c74r6   7.82  -1190. 2399. 2435.    4008.         472.
17 m2c81r6   8.40  -1679. 3377. 3416.   30705.         472.
18 m2c82r7   5.46  -1718. 3450. 3476.  104886.         405.
19 m2c91r6   5.73  -2090. 4193. 4221.  169888.         474.
20 m2c93r6   7.56  -1582. 3181. 3216.   20465.         472.
```

```r
gam_model %>% droplevels() %>% tidy(model, parametric = T)
```

```
# A tibble: 40 x 6
# Groups:   ID [20]
   ID       term        estimate std.error statistic   p.value
   <fct>    <chr>          <dbl>     <dbl>     <dbl>     <dbl>
 1 m1c112r4 (Intercept)    85.0      0.829    102.   1.68e-195
 2 m1c112r4 Choice1        -9.35     1.30      -7.20 8.32e- 12
 3 m1c118r5 (Intercept)    54.2      0.746     72.7  5.51e-220
 4 m1c118r5 Choice1        -1.91     1.24      -1.54 1.24e-   1
 5 m1c128r6 (Intercept)     7.51     0.356     21.1  1.34e- 55
 6 m1c128r6 Choice1        -1.60     0.622     -2.57 1.08e-   2
 7 m1c156r6 (Intercept)   116.       0.870    134.   1.95e-249
 8 m1c156r6 Choice1        -4.56     1.39      -3.27 1.21e-   3
 9 m1c163r6 (Intercept)    32.9      0.316    104.   0.
10 m1c163r6 Choice1        -4.92     0.519     -9.47 1.36e- 19
# ... with 30 more rows
```

```r
#plot for gam fit
ggplot(d.cells, aes(x = Stimulus, y = Spikes, color = Choice)) +
  geom_point(size = 0.5) + facet_wrap(~ID, ncol=5) +
  geom_smooth(method = 'gam', formula = y ~ 1 + s(x, bs="cs")) +
  scale_x_continuous("Stimulus (BinocularR * DisparityFar)") +
  scale_y_continuous("Spikes (spikes/time)")+
  scale_color_discrete(name = "Choice", labels = c("Near", "Far")) +
  labs(title="GAM fit of the relationship between Stimulus, Choice, and Spike rates") +
  theme(plot.title = element_text(hjust=0.5))
```



Q6: Is linear model biased?

Comparing the AIC and BIC between the linear model and the GAM model, some cells' AIC and BIC are similar between the two models but others are not (this depended on the monkey, cell, and run; refering back to the previous visualization, some cells does show more of a linear trend than others). Examining the estimated coeffient values for *Choice* between the linear model and the GAM model it shows that these values are similar. For example, for ID m1c112r4, *Choice* coefficent is *-9.35* for the GAM model, and *-10.12* for the linear model. This may imply that although the linear model does not have a good fit, it may also not necessarily be biased.

Q7: Which cells differ most strongly depending on which of the models?

Cells: m2c82r7, m2c147r7, m2c93r6, m2c137r5, m2c152r7

GAM would fit better for these cells than a linear model, thus if we use a liner model/GAM model there will be a larger difference in effect of Choice.

Session information

```
- Session info ----------------------------------------------------------
 setting  value
 version  R version 3.6.1 (2019-07-05)
 os       macOS High Sierra 10.13.6
 system   x86_64, darwin15.6.0
 ui       X11
 language (EN)
 collate  en_US.UTF-8
 ctype    en_US.UTF-8
 tz       America/New_York
 date     2019-10-24

- Packages --------------------------------------------------------------
 package     * version  date       lib source
 assertthat    0.2.1    2019-03-21 [1] CRAN (R 3.6.0)
 backports     1.1.4    2019-04-10 [2] CRAN (R 3.6.0)
 broom       * 0.5.2    2019-04-07 [1] CRAN (R 3.6.0)
 callr         3.3.1    2019-07-18 [2] CRAN (R 3.6.0)
 cellranger    1.1.0    2016-07-27 [2] CRAN (R 3.6.0)
 cli           1.1.0    2019-03-19 [2] CRAN (R 3.6.0)
 colorspace    1.4-1    2019-03-18 [2] CRAN (R 3.6.0)
 crayon        1.3.4    2017-09-16 [2] CRAN (R 3.6.0)
 desc          1.2.0    2018-05-01 [1] CRAN (R 3.6.0)
 devtools      2.2.0    2019-09-07 [1] CRAN (R 3.6.0)
 digest        0.6.20   2019-07-04 [2] CRAN (R 3.6.0)
 dplyr       * 0.8.3    2019-07-04 [1] CRAN (R 3.6.0)
 DT            0.8      2019-08-07 [1] CRAN (R 3.6.0)
 ellipsis      0.2.0.1  2019-07-02 [2] CRAN (R 3.6.0)
 evaluate      0.14     2019-05-28 [2] CRAN (R 3.6.0)
 fansi         0.4.0    2018-10-05 [2] CRAN (R 3.6.0)
 forcats     * 0.4.0    2019-02-17 [2] CRAN (R 3.6.0)
 fs            1.3.1    2019-05-06 [2] CRAN (R 3.6.0)
 generics      0.0.2    2018-11-29 [2] CRAN (R 3.6.0)
 ggplot2     * 3.2.1    2019-08-10 [2] CRAN (R 3.6.0)
 glue          1.3.1    2019-03-12 [2] CRAN (R 3.6.0)
 gridExtra     2.3      2017-09-09 [2] CRAN (R 3.6.0)
 gtable        0.3.0    2019-03-25 [2] CRAN (R 3.6.0)
 haven         2.1.1    2019-07-04 [2] CRAN (R 3.6.0)
 hms           0.5.1    2019-08-23 [2] CRAN (R 3.6.0)
 htmltools     0.3.6    2017-04-28 [2] CRAN (R 3.6.0)
 htmlwidgets   1.3      2018-09-30 [1] CRAN (R 3.6.0)
 httr          1.4.1    2019-08-05 [2] CRAN (R 3.6.0)
 jsonlite      1.6      2018-12-07 [2] CRAN (R 3.6.0)
```

```
 knitr         1.24     2019-08-08 [2] CRAN (R 3.6.0)
 labeling      0.3      2014-08-23 [2] CRAN (R 3.6.0)
 lattice       0.20-38  2018-11-04 [2] CRAN (R 3.6.1)
 lazyeval      0.2.2    2019-03-15 [2] CRAN (R 3.6.0)
 lemon       * 0.4.3    2019-01-08 [1] CRAN (R 3.6.0)
 lubridate     1.7.4    2018-04-11 [2] CRAN (R 3.6.0)
 magrittr    * 1.5      2014-11-22 [1] CRAN (R 3.6.0)
 MASS        * 7.3-51.4 2019-03-31 [2] CRAN (R 3.6.1)
 Matrix        1.2-17   2019-03-22 [2] CRAN (R 3.6.1)
 memoise       1.1.0    2017-04-21 [1] CRAN (R 3.6.0)
 mgcv        * 1.8-29   2019-09-20 [1] CRAN (R 3.6.0)
 modelr        0.1.5    2019-08-08 [2] CRAN (R 3.6.0)
 munsell       0.5.0    2018-06-12 [2] CRAN (R 3.6.0)
 nlme        * 3.1-140  2019-05-12 [2] CRAN (R 3.6.1)
 pillar        1.4.2    2019-06-29 [2] CRAN (R 3.6.0)
 pkgbuild      1.0.5    2019-08-26 [1] CRAN (R 3.6.0)
 pkgconfig     2.0.2    2018-08-16 [2] CRAN (R 3.6.0)
 pkgload       1.0.2    2018-10-29 [1] CRAN (R 3.6.0)
 plyr          1.8.4    2016-06-08 [2] CRAN (R 3.6.0)
 ppcor       * 1.1      2015-12-03 [1] CRAN (R 3.6.0)
 prettyunits   1.0.2    2015-07-13 [2] CRAN (R 3.6.0)
 processx      3.4.1    2019-07-18 [2] CRAN (R 3.6.0)
 ps            1.3.0    2018-12-21 [2] CRAN (R 3.6.0)
 purrr       * 0.3.2    2019-03-15 [1] CRAN (R 3.6.0)
 R6            2.4.0    2019-02-14 [2] CRAN (R 3.6.0)
 Rcpp          1.0.2    2019-07-25 [2] CRAN (R 3.6.0)
 readr       * 1.3.1    2018-12-21 [2] CRAN (R 3.6.0)
 readxl        1.3.1    2019-03-13 [2] CRAN (R 3.6.0)
 remotes       2.1.0    2019-06-24 [1] CRAN (R 3.6.0)
 rlang         0.4.0    2019-06-25 [2] CRAN (R 3.6.0)
 rmarkdown     1.15     2019-08-21 [2] CRAN (R 3.6.0)
 rprojroot     1.3-2    2018-01-03 [2] CRAN (R 3.6.0)
 rstudioapi    0.10     2019-03-19 [2] CRAN (R 3.6.0)
 rvest         0.3.4    2019-05-15 [2] CRAN (R 3.6.0)
 scales        1.0.0    2018-08-09 [2] CRAN (R 3.6.0)
 sessioninfo   1.1.1    2018-11-05 [1] CRAN (R 3.6.0)
 stringi       1.4.3    2019-03-12 [2] CRAN (R 3.6.0)
 stringr     * 1.4.0    2019-02-10 [2] CRAN (R 3.6.0)
 testthat      2.2.1    2019-07-25 [1] CRAN (R 3.6.0)
 tibble      * 2.1.3    2019-06-06 [2] CRAN (R 3.6.0)
 tidyr       * 0.8.3    2019-03-01 [2] CRAN (R 3.6.0)
 tidyselect    0.2.5    2018-10-11 [2] CRAN (R 3.6.0)
 tidyverse   * 1.2.1    2017-11-14 [2] CRAN (R 3.6.0)
 usethis       1.5.1    2019-07-04 [1] CRAN (R 3.6.0)
 utf8          1.1.4    2018-05-24 [2] CRAN (R 3.6.0)
 vctrs         0.2.0    2019-07-05 [2] CRAN (R 3.6.0)
 withr         2.1.2    2018-03-15 [2] CRAN (R 3.6.0)
 xfun          0.9      2019-08-21 [2] CRAN (R 3.6.0)
 xml2          1.2.2    2019-08-09 [2] CRAN (R 3.6.0)
 yaml          2.2.0    2018-07-25 [2] CRAN (R 3.6.0)
 zeallot       0.1.0    2018-01-28 [2] CRAN (R 3.6.0)

[1] /Users/yinglin/Library/R/3.6/library
[2] /Library/Frameworks/R.framework/Versions/3.6/Resources/library
```